應用位置感知與混合性網路服務之即時行動資訊系統

Applying Location-Aware and Mashup Approach To Design A Real-Time Mobile Information System

陳志達
南台科技大學資管所
andypony@mail.stut.edu.tw

陳詠霖
南台科技大學資管所
m9590103@webmail.stut.edu.tw

摘 要

本研究提出並建構一個以「混合性網路服務」（mashup）為基礎的資訊平台，並且說明了如何以「用戶端混合服務模式」來結合不同來源資訊與服務。一般的行動使用者在透過行動通訊設備查詢地理資訊時，往往需要與自身所處位置相關的資訊內容，因此我們在研究過程中設計並建製了一個資訊平台，由使用者提供地理位置上各種類型的 POI（Point of Interesting）資訊，如熱門的風景名勝、旅遊景點、公共設施等，透過電子地圖服務工具將這些具有主題意義的地點標示出來，並利用 *mashup* 的概念結合第三方網站所提供之網路服務、網路應用程式介面與其他資訊內容，透過 Ajax 架構的使用來輔助使用者操作，讓系統能夠提供交通狀況、天氣預測、實景圖片與鄰近地點查詢的加值 POI 資訊，解決單機軟體所提供的資料無法即時更新的時效性問題，以及地域性資訊缺乏的缺點。

關鍵詞：混合性網路服務，網路服務，網路應用程式介面, Ajax架構, Web 2.0

Abstract

In this paper we propose a *Mashup* Mobile Information System to combine different sources of information and services by *Mashup* approach. Today, most mobile users often operate the mobile equipments, and inquire geographic information with location-related contents they need. The purpose of this research is to provide the POI information about the real geographic location, such as the popular tourist spots, tourist scenery and public facilities. This system also utilizes Web mapping technology like Google Maps to mark the meaningful locations spot. We use the concept of *Mashup* combining Web service, Web API and other information from third-party websites to provide user various POI information. These POIs Include traffic conditions, weather forecasts, pictures and other POI near the location. We also solve the problems that information provided by general software can not be updated timeliness and the disadvantage of lack of geographical information.

Key words：Mashup, Web service, Web API, Ajax framework, Web 2.0

一、前言

根據網際網路的使用經驗，可以發現 Web 在網際網路中所扮演的是資訊共享平台的重要角色，隨著無線網路技術的進步以及行動通訊設備的普及化，讓使用者在

透過網際網路擷取資訊時，不再受到空間與時間的限制，也讓無所不在的運算環境得以實現。

Eija Kaasinen [5]在位置感知服務的使用者需求研究中提出，五個滿足行動使用者需求的要素，分別是：使用者的態度（User attitudes）、資訊與內容（Contents）、使用者與系統的互動性（Interaction）、個人化（Personalization）與順暢的服務(Seamless service entities)，其中一個最重要的層面就是資訊內容。

而隨著 Web2.0 技術與應用的持續演化，讓資訊能更容易地在網際網路上分享及散佈，其主要技術如 Web Service [8]、RSS [23]、Ajax [13]，應用方面較廣為人知的網站如 Wikipedia [24]、flickr [15]及 Google Maps [16]等皆提供了使用者分享與參與的平台或是簡單易用的網路應用程式介面（Web API），更促使一個能夠結合不同來源網站其服務及內容的概念被提出，我們稱之為 *mashup* [4]。

*mashup*指的是混合多種網路應用的網站，能夠將其他網站所提供的服務或內容整合起來並產生新的加值資訊，其資料來源通常是第三方提供的API、RSS或Web service，透過*mashup*的方式，網站開發者能夠很快地結合不同來源網站的服務，並整合形成一個新的網路應用。*mashup*主要的運作方式及其兩種類型將在下一章做詳細說明。

本研究之目的即希望運用 *mashup* 的概念，將各種類型的 POI（Point of Interesting）[21]資訊，例如：風景名勝、旅遊景點、公共設施等，透過視覺化的電子地圖服務將這些具有主題意義的地點標示出來，除了可提供使用者明確了解 POI 的地理位置之外，系統還可根據 POI 的位置提供額外的資訊及服務，像是其本身的簡介資料、所處位置之交通狀況、天氣預測與鄰近地點資訊查詢等，能夠解決單機軟體資料無法即時更新及地域性資訊缺乏的缺點。

因此在本研究論文中，我們除了詳細說明兩種 *mashup* 模式的運作過程之外，也仔細描述出如何運用 Web2.0 技術及其概念，結合第三方服務網站所提供的工具與內容，透過 *mashup* 的方法混合出包含更多資訊及內容的 POI。而在室外的行動使用者也可使用 GPS 接收器，配合行動計算設備與無線網路基礎設施連線至系統，由系統解析其所在位置達到位置感知的目的，使得系統能夠提供使用者更多真實地理環境的各種資訊。

二、相關研究探討

此章節將介紹本研究主要的 *mashup* 概念，以及系統所使用的幾個元件，包括系統使用的 Ajax 架構－ZK、Web Service 技術、第三方應用程式介面、位置感知服務等相關研究工作。

（一）Mashup 的定義與運作模式

Sun 的 Ed Ort, Sean Brydon 與 Mark Basler 在 ”*Mashup Style*” [8] 文章中對*mushup*做了簡單的定義：假設一個網站使用其他網站提供的資料或服務，或是它能存取其他網站所提供的RSS feed並結合成一個新的應用，便可稱為個*mashup*網站。

另外在文章中提出兩種*mashup*的主要模式為：

- 伺服端混合服務（server-side mashups），在此模式中所有從用戶端發出的請求需先經由伺服端的一個代理模組，負責處理使用者的請求並與其他網站進行互動，在整合這些請求的服務及內容後

轉換成一個適合的資料格式呈現給用戶端使用者，其*mashup*的過程是在伺服端所進行。

- 用戶端混合服務（client-side mashups），這種模式少了代理模組的輔助，而是由用戶端直接向第三方網站請求所需要的服務或內容，其*mashup*過程則是在用戶端進行。

在了解兩種*mashup*模式的差異後，接著描述此兩種運作流程。下圖1為伺服端混合服務模式運作圖。

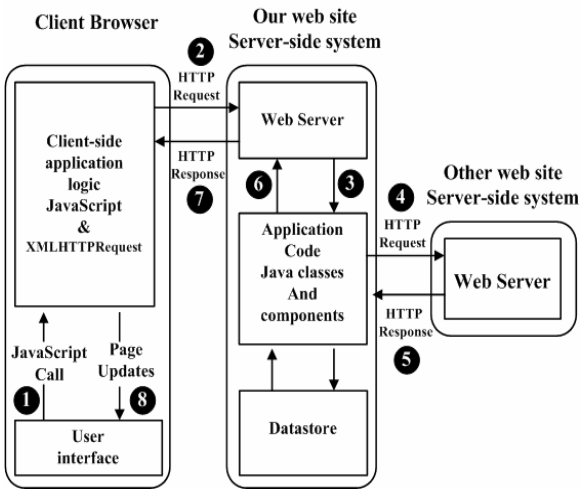

圖1 伺服端混合服務模式運作圖

伺服端混合服務模式運作流程為：

1. 用戶端使用者透過瀏覽器在網頁產生一個事件，並觸發網頁上的JavaScript函式。
2. 用戶端發出請求至我們的網站伺服器。
3. 網站伺服器上的元件接收到用戶端的請求，並呼叫一個或多個代理模組(proxy class)連線至網站所*mashup*的第三方服務網站。
4. 網站的代理模組建立連線至第三方網站，並向其請求用戶端所需要的服務。
5. 第三方服務網站接收到請求，處理並回傳資料給網站的代理模組。通常是使用HTTP GET與HTTP POST做為資料的傳送形式。
6. 網站的代理模組接收回傳資料，並將資料轉換成適合用戶端的資料格式；除此之外，代理模組亦能暫存第三方網站的回傳資料，以進行下一步的用戶端請求處理。
7. 網站將資料回傳給用戶端。
8. 用戶端的回收（callback）函式將回傳的資料更新至使用者的瀏覽頁面。

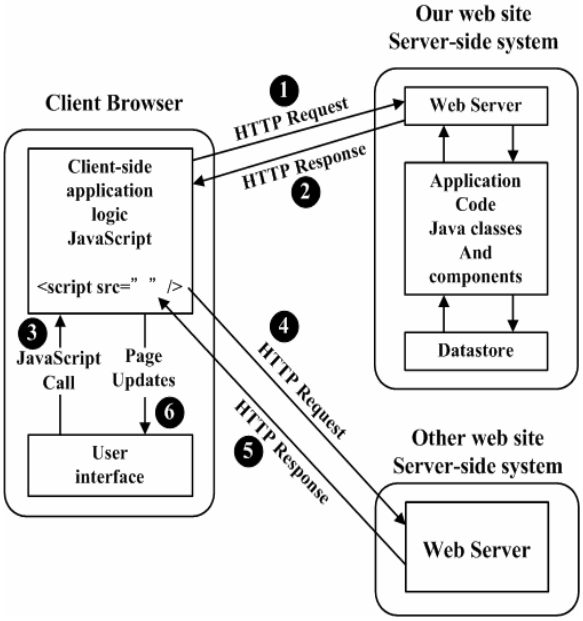

圖2 用戶端混合服務模式運作圖

圖2為用戶端混合服務模式運作圖，其運作流程為：

1. 用戶端使用者透過瀏覽器發出請求至我們的網站伺服器。
2. 瀏覽器從網站伺服器載入一個網頁，該網頁已包含第三方服務網站的JavaScript函式庫（JavaScript library）。舉例來說，假設我們希望在我們的網站中提供使用者能夠使用Google Maps的服務，則必需在網頁中包含Google Maps的JavaScript函式庫一起與網頁載入到用戶端。
3. 用戶端載入網頁後，使用者執行的某些動作即會呼叫JavaScript函式庫中的特定函式，經由函式的處理來動態產生<Script>標籤。

4. 用戶端透過這些<Script>連線至我們所*mashup*的第三方服務網站，並送出HTTP request請求第三方網站提供服務。
5. 第三方服務網站處理請求並回傳資料。
6. 用戶端函式庫中的回收函式將回傳的資料更新至使用者的瀏覽頁面。

（二）Ajax 與 ZK

Ajax（Asynchronous JavaScript and XML）[13]是一種基於 Web 瀏覽器的應用程式，它並不是單一的技術，而是有效地利用一系列的技術，例如使用 DOM [19]來動態地展示資料及互動；使用 XML 進行資料交換；運用 XMLHttpRequest [13]物件來非同步地擷取資料；最後由 JavaScript 將這些技術結合形成一個更有力的應用。Ajax 可以僅向伺服端發送並取回所需要的資料，並在用戶端使用 JavaScript 處理來自伺服端的回應，讓伺服端和用戶端之間交換的資料大量減少，所以使用者能從伺服端得到更快速的回應。

ZK [27]是由 Potix 公司所開發的 Ajax 框架，是一套以 Ajax、XUL [25]以及 Java 為基礎的網頁應用程式開發框架。其包含了一個以 Ajax 為基礎、事件驅動及高互動性的引擎，提供豐富、可重複使用的 XUL 與 HTML 組件，也提供了 XML 為基礎的使用者介面標籤語言 ZUML（ZK User-interfaces Markup Language）[27]，其為一種類似 HTML 的標籤語言，能設計出豐富的使用者介面，讓開發人員更快速組合出多元的 Web 應用。

（三）Web Service 與 SOAP

Web Service [4]提供一種通用的服務，使用遠端程序呼叫（RPC）的方式，透過網際網路建立一個自動機制，直接與位在不同位置的其它應用程式進行資料交換或資源的共享，透過標準的通訊協定，讓用戶端和伺服端能夠依據相同的資料格式和規格來傳遞資訊。

SOAP（Simple Object Access Protocol）[4]是一種標準化的通訊規範，主要用於網路服務（Web Service）中，SOAP 的出現是為了簡化網頁伺服器在從 XML 資料庫中提取資料時，無需花時間去格式化頁面，並能夠讓不同應用程式之間透過 HTTP 通訊協定，以 XML 格式互相交換彼此的資料，如圖 3 所示。

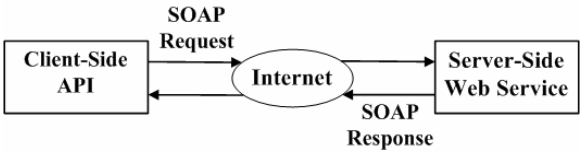

圖 3 使用 SOAP 的 Web Service 架構

（四）第三方應用程序介面與 RSS

Google Maps API 與 Google AJAX Feed API [17]是 Google 提供的網路服務公用介面。Google Maps 提供了各種地圖服務，透過 Ajax 技術的運用，使用者可在瀏覽器上平順的觀看世界各地的地圖影像，而 Google Maps API 可讓開發者將 Google Maps 嵌入在個人網站中，API 本身提供的功能也能讓開發人員設計出更豐富的地圖服務；Google AJAX Feed API 允許開發人員混合（mashup）第三方網站所提供的 RSS feed，除了一般的新聞之外，還能從一些熱門相簿網站例如：Flickr、Picasa Web Albums 所提供的 RSS feed，將網路相簿的圖片嵌入並呈現在個人網頁上。

RSS（Really Simple Syndication）[23]，是一種用來散佈和匯集 Web 內容的 XML 共通格式，常用於新聞網站、部落格（blog）或其他 Web 內容的資料交換，透過 RSS 的使用，可以很容易地產生並傳播資訊內容的鏈結、標題和描述等資料。

（五）位置感知的方式

底下介紹幾種主要的位置感知的方式[7]：全球定位系統（GPS, Global Position System）、紅外線系統（Infrared-based Active Badgesystem）、無線區域網路（Wi-Fi）等其他無線電波的感測方式。

本研究使用 GPS 進行位置感知，GPS 是結合衛星和無線技術，應用在導航定位的導航系統能提供使用者精確的定位、速度及時間，在室外的使用者可利用 GPS 接收器接收衛星的訊號以達到定位的目的，並且能夠在地球的任何地方使用，但若在室內或高樓密集的都市裡使用，其位置感知的效果較差，並且需要較長的啟動及定位時間。所以當行動使用者持有行動計算設備時，配合 GPS 訊號接收器，並由用戶端的 GPS 訊號讀取程式將資料傳送至伺服端系統進行使用者位置的解析，達成位置感知的目的。

GPS 接收器所接收到的的訊號規格是符合 NMEA-0183 [20]的標準，這個標準包含了四種協定，然而在實作上只需要使用者的位置相關資料如經緯度來做應用，因此我們僅需使用 NMEA-0183 規格中的 GPRMC[20]協定中的資料來做為位置判定的依據。GPRMC 訊息的資料格式如：*$GPRMC,232412,A,2202.6921,N,12131.0017,E,000.6,054.9,111106,,,A*74*，表 1 說明了 GPRMC 協定之格式與範例描述。

表 1　GPRMC 協定之格式與範例

Name	Example	Description
Message ID	$GPRMC	RMC protocol header
UTC Time	232412	hhmmss.sss
Status	A	A = Data valid or V = data not valid
Latitude	2202.6921	ddmm.mmmm
N/S indicator	N	N = north or S = south
Longitude	12131.0017	dddmm.mmmm
E/W Indicator	E	E = East or W = West
Speed Over Ground	000.6	Speed Over Ground
Course Over Ground	054.9	True
Date	111106	ddmmyy
Checksum	*74	Checksum

三、系統規劃與設計

我們以用戶端混合服務模式的方式，建構一個整合系統本身資料與第三方網路服務的行動資訊系統，而系統結合的第三方網路服務包括：Google Maps、Google AJAX Feed，第三方之資訊內容包括：Yahoo! Weather [26] 提供的氣象預報資訊，與數位圖片網站 Flickr 所提供之 RSS feed。

以網站開發者的觀點來看，「伺服端混合模式」與「用戶端混合模式」這兩種建構 *mashup* 網站的方法皆有其優缺點，要選擇使用何種模式來建構系統則應先評估此二種方法所能達到的利益與適合的環境及狀況。而在本論文中，我們選擇用戶端混合模式來建構系統的理由及其優點是：

- 用戶端混合模式可以很容易地實現。如果我們希望在自己的網站中加入第三方網站提供的服務，只需將該服務的 JavaScript 函式庫包含在我們的網頁中，網站開發者藉由運用這些函式，即可很簡單地完成混合服務的動作。
- 在用戶端混合模式中所結合的第三方網站皆以提供伺服端元件，但是伺服端混合服務模式則必需額外建置一個伺服端的代理模組做為用戶端與服務之間的媒介。
- 在伺服端混合模式中，一個服務的請求首先要由用戶端瀏覽器發出至伺服端

代理模組，接著再由此代理模組建立連線至第三方網站請求服務，資料的回傳也需要建立相同的兩次連線，這會導致回傳資料時產生較長的延遲時間。然而在用戶端混合模式中，服務的請求與回應是在用戶端與第三方網站之間直接的傳送，所以一般來說，用戶端在接收到回傳資料時只需等待較短的延遲時間。

- 因為用戶端混合模式的服務請求方式無需透過代理模組的仲介，而是由用戶端直接向第三方網站聯繫，可以減低網站伺服器的負擔。
- 大多數的瀏覽器都有提供用戶端混合模式所要的技術支援，網站開發者不需要針對各種不同瀏覽器撰寫客製化的外掛程式來實現這種 *mashup* 的方法。

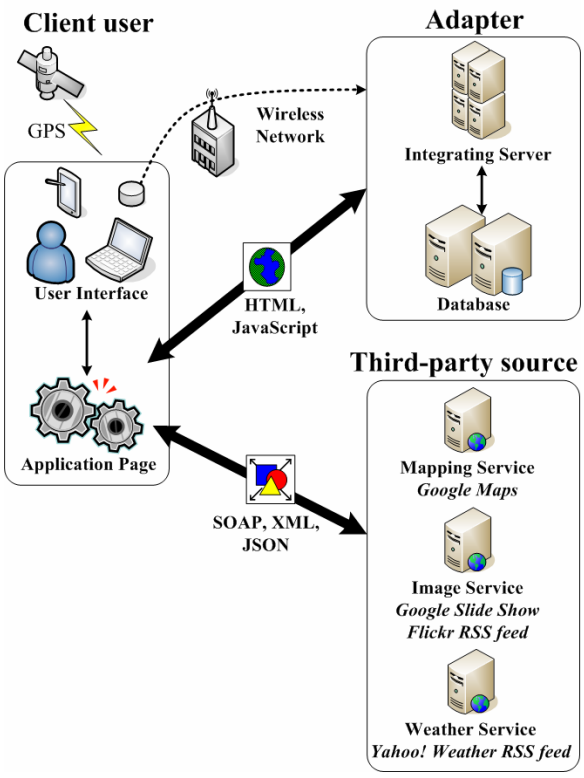

圖 4 以用戶端混合服務為基礎的運作概觀

在圖 4 的系統運作概觀圖中，用戶端的一般使用者或行動使用者，使用瀏覽器對系統發出請求（例如：查詢某城市地圖），系統在收到請求後回應用戶端一個應用網頁，在該應用網頁中已包含了使用者介面與第三方服務網站提供的 JavaScript 函式（Google Maps API），使用者透過瀏覽器在應用頁面中執行動作時，便會觸發函式呼叫第三方網站（Google Maps）來提供服務。第三方網站在讀入用戶端的請求語法（Script）後，用戶端瀏覽器上的應用頁面即會開始執行回收（Callback）的程序，接收由第三方網站回傳的參數，接著解析回傳的參數並把結果更新並顯示在用戶端網頁上，完成使用者的請求。構成系統運作的幾個主要角色為：

- 用戶端使用者（Client user）：使用者利用瀏覽器透過網際網路與我們的系統互動，並直接與第三方網站請求服務與資料傳輸；室外的使用者亦可藉由行動設備與 GPS 接收器，配合無線網路基礎設施與系統溝通達到位置感知的目的，進而能提供行動使用者了解所處地理位置的周遭資訊。
- 適配系統（Adapter）：主要由一個整合伺服器與資料庫組成。這個角色的主要工作是提供用戶端連接第三方網站所需要的 JavaScript 函式庫，以及系統所使用的 Ajax 函式庫：ZK。除此之外適配系統也結合了系統本身提供的相關服務包括：位置感知、氣象預測的 Web Service、POI 資訊等服務。
- 第三方來源服務（Third-party source）：系統所 *mashup* 的第三方網站服務包括 Google Maps、Google Slide Show、Flickr RSS feed、Yahoo! Weather RSS feed。透過 Google Maps 的使用，我們能夠將各個 POI 資訊呈現在視覺化的電子地圖上，系統使用者可以明確了解這些地圖的詳細位置；藉由 Google Slide Show 的運用，以及網路相簿如

Flickr 所提供的 RSS feed，我們的系統允許使用者能夠將他們在網路相簿中的圖片結合至 POI 資訊中；Yahoo! Weather RSS feed 則提供了全世界各主要城市的氣象資訊。我們希望利用這些不同的服務及資源來增強 POI 內容，以創造出更健全的地圖應用。

四、系統運作組成與功能說明

為實現本研究提出的 *mashup* 行動資訊系統，我們使用 ZK 的版面組件，配合 HTML 與 JSP，設計並製作一個簡單易用的使用者介面。使用者介面可分為兩個部分，在圖 5 中的左半邊為 *mashup* 資訊的呈現部份，混合了包括 POI 的地圖展現、詳細資訊、氣象預測、圖像資訊等，透過分頁標籤的點選，使用者能很輕易且迅速的得到 POI 的相關資訊；右半部份為 POI 資訊的搜尋介面，提供使用者可依熱門標籤、區域別、功能別、關鍵字等方式來搜尋 POI 資訊，搜尋是透過 Ajax 引擎來存取伺服端資料庫的資料，減少使用者在搜尋資料時從系統重新載入網頁的時間。

圖 5 瀏覽器上的用戶端使用者介面

（一）使用者介面

本系統主要使用 ZK 的 UI 物件來建製使用者介面，在網頁中加入 ZUML 的名稱空間定義之後，即可在 HTML 網頁中插入 ZUML 標籤語言，能豐富用戶端網頁所呈現出的使用者介面。圖 6 為使用 ZK 分頁標籤的 ZUML 範例，其功能可讓網頁透過分頁標籤來呈現大量資料內容；圖 7 為系統處理 ZUML 後在網頁上呈現的結果。

```
<z:tabbox width="400px">
    <z:tabs>
        <z:tab label="Tab 1"/>
        <z:tab label="Tab 2"/>
    </z:tabs>
    <z:tabpanels>
        <z:tabpanel>This is panel one</z:tabpanel>
        <z:tabpanel>This is panel two </z:tabpanel>
    </z:tabpanels>
</z:tabbox>
```

圖 6　ZUML 的分頁標籤範例

圖 7　圖 6 的 ZUML 執行結果

（二）POI 資訊的展現

在之前的章節中提到，POI 資訊可提供使用者熱門的地標或景點資訊，而透過電子地圖服務，可輔助系統將資料庫中的 POI 資料展現在使用者瀏覽器上。在本系統用戶端的使用者介面主要是一個 JSP 網頁，包含了 Google Maps API 的 JavaScript 函式，負責監聽使用者在瀏覽器上的活動與事件，可擷取來自系統資料庫的 POI 資料，例如：地標經緯度、地標名稱、地標描述、網址等，透過 Google Maps 的資訊視窗呈現資料，如下圖 8 所示。

圖 8　使用 Google Maps 展現 POI 資訊

（三）結合圖像 RSS 的應用

隨著網路應用的發展，越來越多使用者會將旅遊過程中所拍攝的照片上傳至網路相簿中，而一些較熱門的網路相簿網站如 Flickr [15]、Picasa [22]也開始提供媒體型態的 RSS feed。

系統使用 Google Ajax Feed API，提供使用者在新增一筆新的 POI 資訊時，除了文字的描述之外，還可結合網路相簿至 POI 資訊中，並讓這些圖像具有滑動效果的展示功能。圖 9 為 Flickr 網路相簿提供的 RSS feed 範例；圖 10 為 Google Ajax Feed API 展示 RSS 的圖例。

```
<rss version="2.0" xmlns:media="http://search.yahoo.com/mrss/"
xmlns:dc="http://purl.org/dc/elements/1.1/">
 <channel>
   <title>yonglin724 的相片</title>
   <link>...</link>
   <pubDate>...</pubDate>
   <lastBuildDate>...</lastBuildDate>
   <generator>http://www.flickr.com/</generator>
   <item>
     <media:content url="..."  type="image/jpeg" />
     <media:title>南台科大圖書館-視聽區</media:title>
     <media:text type="html">...</media:text>
     <media:thumbnail url="..." />
     <media:credit role="photographer">yonglin724</media:credit>
     <media:category scheme="urn:flickr:tags">...</media:category>
   </item>
 </channel>
</rss>
```

圖 9 網路相簿 Flickr 提供的 RSS feed 範例

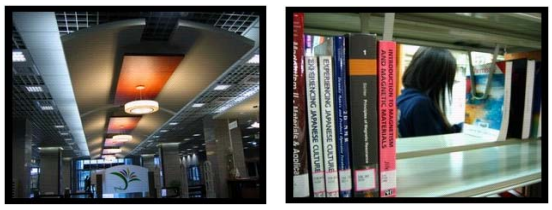

圖 10 Google Ajax Feed API 解析 RSS feed 後呈現之結果

（四）結合氣象資訊

在本研究中，我們實作了一個氣象查詢網路服務（Web Service），利用 Yahoo! Weahter [26]提供的 RSS feed，將台灣各縣市之氣象資訊結合到系統中，讓使用者在查詢 POI 資訊的同時，也能將該地區的氣象資訊提供給使用者。

氣象資訊查詢的網路服務，主要運作環境由 Apache Axis [14]、JWSDP [18]、JDOM [19]建構，當使用者在查詢 POI 資訊時，便會觸發網路服務的剖析模組進行氣象資料查詢動作，這個動作根據使用者所要查詢的地域，剖析 Yahoo! Weather 提供的氣象 RSS feed，最後將查詢結果透過 SOAP 訊息回傳給用戶端使用者，回傳的氣象資訊包括：氣溫、風速、濕度、能見度等，如圖 11 所示。

```
< soapenv:Envelope ... >
 < soapenv:Body >
  <parseResponse ...>
   <parseReturn ...>
     <item>Wed, 18 Jul 2007 3:30 pm CST</item>
     <item>City: T'ai-nan</item>
     <item>Temp: 32 .C</item>
     <item>Speed: 19 Km/h</item>
     <item>Humidity: 62 %</item>
     <item>Visibility: 9990 M</item>
     <item>Sunrise: 5:24 am</item>
     <item>Sunset: 6:47 pm</item>
   </parseReturn>
  </parseResponse>
 </ soapenv:Body >
</ soapenv:Envelope >
```

圖 11 Web Service 處理氣象資訊查詢動作後回傳至用戶端的 SOAP 內容

（五）社會性網路的概念與應用

社會性網路服務的目的，主要是一個以 Web 為基礎，並且提供許多互動的方式，讓使用者能互相交流他們的興趣及活動，建立出一個高度關係的社群網路，而最常見的社會性網路服務，例如：聊天室、部落格、影音分享網站等。本系統也以此做為建構系統的概念，讓使用者能建立屬於自己的個人資訊記錄，其他使用者亦可將系統中的 POI 加入至個人的喜好記錄中。而藉由一些功能像是：評分機制、標籤註解、意見的分享等，讓系統的使用者建立起友誼關係並發展出社群網路，進而

達到 Web2.0 以人與人之間關係的建立與管理之目的。

五、結論與未來發展

在本研究論文中，我們提出了以「用戶端混合服務」的方式來建構一個整合系統本身資料與第三方網路服務的行動資訊系統，利用第三方網站的服務內容，例如：RSS、Web Service，提供使用者有用的相關地理資訊，並將 *mashup* 後產生之 POI 資訊透過簡單易用的介面呈現給使用者。讓系統能夠提供交通狀況、天氣預測、實景圖片與鄰近地點查詢的加值 POI 資訊，解決單機軟體所提供的資料無法即時更新的時效性問題，以及地域性資訊缺乏的缺點。我們也在論文中詳細描述此研究的實作成果與功能，說明了如何結合不同的技術、工具、資訊內容，並將它們轉換成一個實用且可行的 Web 應用。

在未來的工作中，將會加入更便利的使用者介面以及更多實用的資訊服務，例如：當使用者在查詢某個車站的 POI 資訊時，系統便會主動提供使用者各級列車資訊；或是結合影音服務至 POI 資訊中。另一方面，可加入語意網路的概念，將各個 POI 定義出語意關係，以實現更具智慧與更準確的資料搜尋。

六、參考文獻

[1] 姚岦昕、蔡尚榮，引用網絡服務技術之位置感知資訊服務，國立成功大學電腦與通信工程所。

[2] Benjamin N Lee, Wen-Yen Chen, Edward Y. Chang, “A Scalable Service for Photo Annotation, Sharing, and Search”, Proceedings of the 14th annual ACM international conference on Multimedia '06.

[3] Feng Jing, Lei Zhang, Wei-Ying Ma, “ VitualTour: An online Assistant Based on High Quality Images”, Proceedings of the 14th annual ACM international conference on MULTIMEDIA '06.

[4] Harry Halpin, Henry S. Thompson, “combining XML, web services, and the semantic web,” Proceedings of the 15th international conference on World Wide Web '06, SESSION: XML & web services, pp. 679 – 686.

[5] Kaasinen,“User needs for location-aware mobile services”, Personal and Ubiquitous Computing, volume 7, issue 1, pp.70-79, 2003.

[6] Mira Dontcheva, Steven M. Drucker, Geraldine Wade, David Salesin, Michael F. Cohen, “Summarizing personal web browsing sessions,” Proceedings of the 19th annual ACM symposium on User interface software and technology UIST '06, SESSION: Browsing & scrolling, pp.115-124.

[7] Mike Hazas, James Scott, John krumm, “Location-Aware Computing comes of Age”, Computer, IEEE Computer Sopciety, pp.95-97, Feb. 2004.

[8] Mashup Styles, Part 1: Server-Side Mashups By Ed Ort, Sean Brydon, and Mark Basler, May 2007. http://java.sun.com/developer/technicalArticles/J2EE/mashup_1/index.html

[9] Mashup Styles, Part 2: Client-Side Mashups By Ed Ort, Sean Brydon, and Mark Basler, May 2007. http://java.sun.com/developer/technicalArticles/J2EE/mashup_1/index.html

[10] Sudarshan Murthy, David Maier, Lois Delcambre,“Mash-o-matic”, Proceedings

of the ACM symposium on Document engineering ’06.

[11] Sayar A, Pierce M, Fox G, “Integrating AJAX Approach into GIS Visualization Web Services”,Indiana University, USA.

[12] Yuuichi Teranishi, Junzo Kamahara, Shinji Shimojo, “MapWiki: A Ubiquitous Collaboration Environment on Shared Maps”, Proceedings of the International Symposium on Applications on Internet Workshops - Volume 00 SAINT-W '06 IEEE.

[13] Ajax: A New Approach to Web Appl ications http://www.adaptivepath.com/p ublications/essays/archives/000385.php

[14] Apache Axis. http://ws.apache.org/axis/

[15] Flickr. http://www.flickr.com/

[16] Google Maps. http://maps.google.com

[17] Google APIs. http://code.google.com/

[18] JWSDP. https://metro.dev.java.net/

[19] JDOM. http://www.jdom.org/

[20] NMEA-0183,http://www.nmea.org/pub/ 0183/

[21] POI, http://en.wikipedia.org/wiki/Point_ of_interest

[22] Picasa. http://picasaweb.google.com/

[23] RSS Specification. http://www.rssboard. org/

[24] Wikipedia. http://www.wikipedia.org

[25] XUL. http://xul.sourceforge.net/mozilla. html

[26] Yahoo! Weather. http://weather.yahoo.c om/

[27] ZK. http://www.zkoss.org/